OPT2025: 17th Annual Workshop on Optimization for Machine Learning

# Augmented Normalization: Differentiating the Generalized Geometric Median

**Tyler King** TTK22@CORNELL.EDU
*Cornell University, New York, United States of America*

**Ser-Nam Lim** SERNAM@UCF.EDU
*University of Central Florida, Florida, United States of America*

## Abstract

We introduce Augmented Normalization (AugNorm), a novel feature transformation method that normalizes around a generalized geometric median (GGM). Unlike traditional normalization techniques that require fixed statistics such as the mean or median, AugNorm formulates the GGM as the minimizer of a convex $\arg\min$ function, enabling a smooth interpolation between generic test statistics such as the mean, median, and range. This yields a noise-robust test statistic that avoids optimization difficulties associated with median-based methods. On CIFAR-10, AugNorm matches BatchNorm in-distribution while outperforming median normalization. On CelebA, a subpopulation shift dataset, we show AugNorm strengthens out-of-distribution robustness by improving worst-case test accuracy. We extend our method by introducing a differentiable variant of AugNorm, where the test statistic becomes a trainable parameter. Our results indicate AugNorm is a simple and effective drop-in replacement for BatchNorm that complements existing robust training schemes for settings with distribution shift.

## 1. Introduction

Normalization techniques have played an integral role in the effective training of deep neural networks, with Batch Normalization (BatchNorm) [20] demonstrating that normalizing activations across mini-batches to zero mean and unit variance can stabilize training and accelerate convergence. While BatchNorm has been widely adopted due to its ability to smooth loss landscapes [36, 43] and improve optimization [3, 22], it also has significant pitfalls [19, 38, 42]. To address these limitations, various modifications [5, 16, 26, 31, 41, 44] and alternative normalization schemes [1, 4, 34, 42] have been introduced, highlighting the importance of normalization in deep learning and the interest in exploring new formulations.

One such area includes modifying the deviation measures for scaling and statistics for centering. Examples include rescaling BatchNorm by the $L^1$-norm [41] or normalizing around the range [44] (which can be considered normalizing via an $L^\infty$-norm). These variations of standardization typically operate orthogonal to task-specific normalization schemes and thus can be painlessly integrated into other normalization blocks [18]. Such operations serve as the theoretical basis for AugNorm, which interpolates between these by considering a generalization of the geometric median (via an exponent $\phi$), which computes the median in the univariate case [12, 30].

One area where varied normalization test statistics could play a key role is in subpopulation shift, where the testing data distribution is a specific or the worst-case subpopulation of the training distribution [45]. Due to its association with numerous tasks such as learning for algorithmic

fairness [2, 7] or learning with class imbalance [10, 21], being able to improve performance under these conditions is of particular interest. Prior work has been focusing on modifying the training framework, replacing empirical risk minimization (ERM) approaches with distributionally robust optimization (DRO) [6, 32], which minimize a model's loss over the worst-case distribution based on a neighborhood of the observed training distribution [45]. Further extensions such as DORO [45] exist, avoiding overfitting to potential outliers that cause poor DRO performance in real-world settings [46].

Despite numerous approaches modifying the optimization step for robust training, we note a lack of prior literature addressing underlying issues in model architecture. We empirically demonstrate that AugNorm serves as a viable drop-in replacement, improving worst-case test accuracy under a variety of DRO-based training procedures.

## 2. Background

### 2.1. Subpopulation Shift

Consider the setting with input space $\mathcal{X}$ and output space $\mathcal{Y}$, where data is drawn from the distribution $P = \mathcal{X} \times \mathcal{Y}$. In addition, each input $x \in \mathcal{X}$ is associated with a collection of attributes $a_1, \ldots, a_K \in \mathcal{A}_1 \times \cdots \times \mathcal{A}_k$, which induce subpopulations $G \in \mathcal{G}$ on the data. The overall distribution $P$ can thus be represented as

$$P = \sum_{g \in \mathcal{G}} P(G = g) P(\mathcal{X}, \mathcal{Y} | G = g). \tag{1}$$

Subpopulation shift occurs when subpopulation mixture differs between training and testing: i.e. $P_{\text{train}}(G) \neq P_{\text{test}}(G)$ while $P_{\text{train}}(\mathcal{X}, \mathcal{Y}|G) = P_{\text{test}}(\mathcal{X}, \mathcal{Y}|G)$. The goal of this task is to train a model $f_\theta : \mathcal{X} \to \mathcal{Y}$ that minimizes the expected risk over the worst-case subpopulation in $\mathcal{G}$.

For the purposes of this paper, all experiments are tested in the overlapping setting, while additionally being subpopulation-oblivious. The former implies that distinct subpopulations are capable of having overlapping data, while the latter implies that the training algorithm is not aware of the subpopulation membership of any instance (including number of subpopulations). These settings are chosen in particular due to their more common occurrence in real-world applications, along with their frequent usage in prior literature [13, 45].

### 2.2. DRO and DORO

Under the subpopulation-oblivious setting we cannot evaluate worst-case group risk since the subpopulations $G \in \mathcal{G}$ are unknown. Instead, Distributionally Robust Optimization (DRO) trains the model to perform well on the worst-case distribution $Q$ that lies within a ball w.r.t. divergence $D$ around the empirical training distribution $P$. More formally, DRO minimizes the expected DRO risk

$$\mathcal{R}_{D,\rho}(\theta; P) = \sup_{Q \ll P} \left\{ \mathbb{E}_Q[\ell(\theta; Z)] \ : \ D(Q \parallel P) \leq \rho \right\} \tag{2}$$

where $\ell(\theta; Z)$ is the loss of model $f_\theta$ on $Z \in \mathcal{X} \times \mathcal{Y}$, $D$ is formulated as the Cressie-Read family of Rényi divergence [13], and $\rho > 0$ controls the robustness region.

DORO [45] further extends this by excluding a small fraction of worst-case samples from the optimization objective, balancing robustness with tolerance to outliers and improving stability in settings with high data imperfection.

### 2.3. Generalized Normalization Methods

A complementary line of work explores alternative normalization strategies that modify the centering and scaling statistics. Examples include rescaling BatchNorm with the $L^1$-norm [41] or normalizing around the $L^\infty$-norm [44]. Closest to our approach, Streaming Normalization [25] rescales activations by an $L^p$ norm, computed from the $p$-th root of the $p$-th absolute moment, but still updates the mean identically to BatchNorm and thus only operates as a rescaled BatchNorm.

Our method requires differentiating through $\arg\min$ operations, for which we leverage techniques from [11]. This enables exact gradient computation during backpropagation, whereas automatic differentiation in PyTorch [33] would be impractical due to long computational graphs arising from iterative solvers such as Newton's method. Explicit gradient derivations are also both faster and more memory-efficient.

Similar to AugNorm, related work has studied differentiating the Fréchet mean [29], which has been applied to hyperbolic normalization on Riemannian manifolds. Since the Fréchet mean also generalizes the geometric median [8, 9] (via a linear combination of terms instead of exponentiation), their approach to gradient computation is similar in style to our derivations.

## 3. Augmented Normalization

Here we define the generalized geometric median (GGM), which is the optimized function, along with the forward and backward passes of the AugNorm block. Assuming Algorithm 1 is applied to images, $d$ is the number of channels $C$ at a specific layer, $m = NHW$ which includes all images in a batch $N$ along with the height $H$ and width $W$, and $\epsilon$ is a small error term to prevent numerical issues.

**Algorithm 1** Forward Pass of AugNorm

**Input** : $\mathbf{x} \in \mathbb{R}^{m \times d}$, learnable parameters $\boldsymbol{\gamma} \in \mathbb{R}^d$, $\boldsymbol{\beta} \in \mathbb{R}^d$, $\boldsymbol{\phi} \in \mathbb{R}^d$

**Output:** $\mathbf{Y} \in \mathbb{R}^{m \times d}$

**for** $j \leftarrow 1$ **to** $d$ **do**

$$\mu_j \leftarrow \arg\min_y \sum_{i=1}^{m} \left| x_{ij} - y \right|^{\phi_j}$$

$$\sigma_j^2 \leftarrow \frac{1}{m} \sum_{i=1}^{m} \left( x_{ij} - \mu_j \right)^2$$

**for** $i \leftarrow 1$ **to** $m$ **do**

$$\hat{x}_{ij} \leftarrow \left( x_{ij} - \mu_j \right) \left( \sigma_j^2 + \epsilon \right)^{-1/2}$$

$$Y_{ij} \leftarrow \gamma_j \, \hat{x}_{ij} + \beta_j$$

**end**

**end**

### 3.1. Generalized geometric median

We flesh out the concept of a generalized geometric median (GGM), which serves as a basis for AugNorm. For $m$ points $x_1, x_2, \dots x_m$, where each $x_i \in \mathbb{R}$, the geometric median in the 1-dimensional case (which coincides with the median in $\mathbb{R}$) is defined as $\arg\min_{y \in \mathbb{R}} \sum_{i=1}^{m} |x_i - y|$.

We generalize this to the formula $\arg\min_{y \in \mathbb{R}} \sum_{i=1}^{m} |x_i - y|^{\phi}$, where $\phi$ is an arbitrary parameter. When $\phi = 1$, our minima $y$ coincides with the median, and when $\phi = 2$, $y$ coincides with the mean.

**Theorem 1** *The landscape of the generalized 1-dimensional geometric median*

$$\sum_{i=1}^{m} |x_i - y|^{\phi} \tag{3}$$

*for $y, x_i \in \mathbb{R}$ is convex with respect to $y$ when $\phi \geq 1$. Specifically, the second partial derivative of this surface has the form*

$$\frac{\partial f(x,y)}{\partial^2 y} = \sum_{i=1}^{m} \frac{(\phi^2 - \phi)|x_i - y|^{\phi}}{(x_i - y)^2} \tag{4}$$

*which is non-negative for $\phi \geq 1$ (proof in Appendix A).*

There does not exist a closed-form equation or algorithm that easily computes $y$ (as there does with the median and mean). However, we can utilize gradient descent, Newton's method, or any other hill climbing algorithm to estimate $y$ when $\phi \geq 1$. In our experiments, we restrict $\phi > 1.2$ due to $\phi \approx 1$ causing stability issues in the forward pass.

### 3.2. Optimizing generalized geometric median

In order to optimize the generalized geometric median, we use Newton's method for efficient calculations. This was due to it being one of the most efficient methods for finding convex minima, which our generalized geometric median observes when $\phi > 1$.

While first-order methods like gradient descent rely on local linear approximations, higher-order methods such as Newton's method yield faster converging algorithms at additional compute cost. Thanks to AugNorm minimizing a univariate function, inverting the Hessian is relatively cheap and led us to use Newton's method instead of alternatives. In our experiments, we found that $\sim$4 iterations were sufficient to achieve `float16` precision.

Additionally, a component of the second order derivative in Newton's method is needed for backpropagation, and thus can save computational costs via caching. Specifically, we need to compute $f_Y$ and $f_{YY}$, the first and second derivative of our generalized geometric median with respect to minima $y$ (their computation can be observed in Eqn. 16). While these values are approximated for in Newton's method (since the input $y$ for $f_Y$ and $f_{YY}$ is not the exact global minima), they still yield low error. Furthermore, since we need both to compute $\frac{\partial y}{\partial x_i}$ for each $x_i$ [11], we can cache $f_{YY}$ from forward propagation and note that $f_{XY}$ naturally follows from $f_Y$.

## 4. Backpropagation

Training a model using an AugNorm layer requires computing gradients with respect to some chosen loss function for backpropagation. Gradients and their derivations are defined similar to the

original BatchNorm paper. We use the multivariate version of chain rule on computational graphs

$$\frac{\partial \ell}{\partial x_i} = \sum_{j \in N(i)} \left( \frac{\partial x_j}{\partial x_i} \right)^T \cdot \frac{\partial \ell}{\partial x_j} \tag{5}$$

where $\frac{\partial \ell}{\partial x_i}$ is the desired gradient with respect to some node $x_i$, $N$ is a function for all descendant nodes of $x_i$, and $\frac{\partial x_j}{\partial x_i}$ is a Jacobian matrix. We derive explicit gradients for backpropagation in Appendix C.

The only nontrivial derivation is computing a gradient across the $\arg\min$ operator (all other derivations are derived similarly to BatchNorm). A closed-form expression exists and a single variable variant is proved in [11]. We annotate the implemented gradients in Theorem 2, which showcases the backward pass in PyTorch.

## 5. Experiments

We first benchmark AugNorm with $\phi = 1.5$ against median normalization and BatchNorm on CIFAR-10, showcasing that AugNorm performs similarly to BatchNorm and additionally showcase the failings of median normalization. We also perform an apples-to-apples comparison between AugNorm and BatchNorm by lifting code from DORO [45] and comparing all training schemes with and without AugNorm, while also testing a differentiable $\phi$ implementation of AugNorm that outperforms both models. For the differentiable $\phi$ implementation, we initialize $\phi = 1.5$ at the start of training.

Additional details for dataset specifics, model architectures and hyperparameters are including in Appendix D.

### 5.1. CIFAR-10 Image Classification

Table 1: Comparison of test accuracies for different models with BatchNorm (BN), AugNorm with $\phi = 1.5$ (AN-1.5), and Median Normalization (equivalent to $\phi = 1$) on CIFAR-10.

| Model | BN | AN-1.5 | Median Norm |
|---|---|---|---|
| ResNet-18 | $95.40 \pm 0.16$ | $\mathbf{95.49 \pm 0.14}$ | $93.51 \pm 0.25$ |
| MobileNet-V2 | $92.40 \pm 0.32$ | $\mathbf{93.11 \pm 0.11}$ | $89.97 \pm 0.20$ |
| DenseNet121 | $\mathbf{95.58 \pm 0.17}$ | $95.49 \pm 0.18$ | $93.77 \pm 0.15$ |
| VGG19 | $\mathbf{93.79 \pm 0.11}$ | $93.77 \pm 0.19$ | $91.18 \pm 0.14$ |
| EfficientNet-B0 | $90.98 \pm 0.24$ | $\mathbf{91.11 \pm 0.17}$ | $90.48 \pm 0.36$ |

Our results indicate AugNorm with $\phi = 1.5$ (AN-1.5) consistently improves or matches BatchNorm (BN) across most architectures, with the largest gain observed on MobileNet-V2, where AN-1.5 boosts accuracy by about 0.7%. In contrast, Median Normalization underperforms BN and AN-1.5 across all tested models, typically trailing by 2–3%. These results suggest that AN-1.5 is a comparable alternative to BN, offering slight improvements to test accuracy. However, Median Normalization trails significantly, likely due to sparse gradients being outputted from the median

(due to only a single input having a non-zero gradient when differentiating through the median operation).

### 5.2. Subpopulation Shift Image Classification

Table 2: Worst-case test accuracies under different normalization methods on CelebA. DiffPhi represents the differentiable $\phi$ implementation. Bold numbers indicate the best result per training procedure. Average results included in Appendix E.

| **Method** | **BN** | **AN-1.5** | **DiffPhi** |
|---|---|---|---|
| ERM | $\mathbf{55.44 \pm 4.03}$ | $53.83 \pm 6.47$ | $55.00 \pm 1.99$ |
| $\chi^2$ | $65.59 \pm 5.96$ | $67.77 \pm 3.41$ | $\mathbf{68.49 \pm 4.68}$ |
| $\chi^2$-DORO | $67.70 \pm 3.41$ | $67.74 \pm 1.88$ | $\mathbf{67.76 \pm 3.16}$ |
| CVaR | $66.80 \pm 3.03$ | $\mathbf{70.48 \pm 4.15}$ | $69.34 \pm 3.77$ |
| CVaR-DORO | $66.28 \pm 3.78$ | $\mathbf{71.72 \pm 2.72}$ | $66.28 \pm 8.09$ |

Because our experiments are lifted from DORO [45], we utilize the optimal hyperparameters observed in their paper and keep the training procedure identical (specifics in Appendix D). Table 2 summarizes worst-case test accuracies on CelebA across five training strategies.

Overall, both AugNorm and DiffPhi noticeably improve robustness across several DRO variants, ranging in performance increases from 3% to 5% on $\chi^2$, CVaR, and CVaR-DORO. While the differentiable $\phi$ implementation yields better performance on $\chi^2$-based architectures, AN-1.5 achieves better performance on CVaR variants. Under ERM, BN performed the best, likely due to both AN-1.5 and DiffPhi biasing the model towards overcompensating outliers (while DiffPhi performed nearly comparable to BN due to $\phi$ values converging towards 2 during training).

These results suggest AugNorm integrates smoothly with existing robust optimization frameworks (as a drop-in replacement for BatchNorm) but can also amplify their benefits under subpopulation shift. This highlights AugNorm's ability to enhance worst-case group performance under subpopulation-oblivious conditions, meshing well with training schemes such as DRO and DORO.

## 6. Conclusion and Future Work

In this paper we introduce AugNorm, a new normalization scheme for distributional robustness. We introduce the generalized geometric median, a novel test statistic, showcasing a way to compute an approximate minimum of this function via Newton's method. We then compute explicit gradients for backpropagation with respect to the various parameters in AugNorm, and finally benchmark our algorithm against common normalization methods such as BatchNorm and AugNorm on both CIFAR-10 and CelebA, achieving similar in-distribution performance while improving worst-case testing accuracy under several training schemes.

In future work, we hope to expand to large and more varied datasets (ex: CivilComments-Wilds [24] and CIFAR-10-C [15]), where we believe AugNorm would also achieve robust performance. We would also like to rigorously benchmark our approach with hyperparameter tuning over $\phi$ (as opposed to a fixed $\phi = 1.5$ implementation) which may further improve test accuracy.

## References


[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[2] Solon Barocas and Andrew D Selbst. Big data's disparate impact. *Calif. L. Rev.*, 104:671, 2016.

[3] Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. *Advances in neural information processing systems*, 31, 2018.

[4] Tianle Cai, Shengjie Luo, Keyulu Xu, Di He, Tie-Yan Liu, and Liwei Wang. Graphnorm: A principled approach to accelerating graph neural network training. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 1204–1215. PMLR, 2021. URL http://proceedings.mlr.press/v139/cai21e.html.

[5] Lucas Deecke, Iain Murray, and Hakan Bilen. Mode normalization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyN-M2Rctm.

[6] John C Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378–1406, 2021.

[7] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, page 214–226. ACM, January 2012. doi: 10.1145/2090236.2090255. URL http://dx.doi.org/10.1145/2090236.2090255.

[8] P. Thomas Fletcher, Suresh Venkatasubramanian, and Sarang Joshi. Robust statistics on riemannian manifolds via the geometric median. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. doi: 10.1109/CVPR.2008.4587747.

[9] P. Thomas Fletcher, Suresh Venkatasubramanian, and Sarang Joshi. The geometric median on riemannian manifolds with application to robust atlas estimation. *NeuroImage*, 45(1):S143–S152, March 2009. doi: 10.1016/j.neuroimage.2008.10.052. URL https://doi.org/10.1016/j.neuroimage.2008.10.052.

[10] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):463–484, 2011.

[11] Stephen Gould, Basura Fernando, Anoop Cherian, Peter Anderson, Rodrigo Santa Cruz, and Edison Guo. On differentiating parameterized argmin and argmax problems with application to bi-level optimization. *CoRR*, abs/1607.05447, 2016. URL http://arxiv.org/abs/1607.05447.

[12] J. B. S. Haldane. Note on the median of a multivariate distribution. *Biometrika*, 35(3-4): 414–417, 1948. doi: 10.1093/biomet/35.3-4.414. URL https://doi.org/10.1093/biomet/35.3-4.414.

[13] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR, 2018.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

[15] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

[16] Xuefeng Hu, Gokhan Uzunbas, Sirius Chen, Rui Wang, Ashish Shah, Ram Nevatia, and Ser-Nam Lim. Mixnorm: Test-time adaptation through online normalization estimation. *arXiv preprint arXiv:2110.11478*, 2021.

[17] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[18] Lei Huang, Jie Qin, Yi Zhou, Fan Zhu, Li Liu, and Ling Shao. Normalization techniques in training dnns: Methodology, analysis and application, 2020.

[19] Sergey Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. *Advances in neural information processing systems*, 30, 2017.

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL http://arxiv.org/abs/1502.03167.

[21] Nathalie Japkowicz. The class imbalance problem: Significance and strategies. In *Proc. of the Int'l Conf. on artificial intelligence*, volume 56, pages 111–117, 2000.

[22] Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. The normalization method for alleviating pathological sharpness in wide neural networks. *Advances in neural information processing systems*, 32, 2019.

[23] Rohan V Kashyap. A survey of deep learning optimizers-first and second order methods, 2022.

[24] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning (ICML)*, 2021.

[25] Qianli Liao, Kenji Kawaguchi, and Tomaso Poggio. Streaming normalization: Towards simpler and more biologically-plausible normalizations for online and recurrent learning, 2016. URL https://arxiv.org/abs/1610.06160.

[26] Hyesu Lim, Byeonggeun Kim, Jaegul Choo, and Sungha Choi. Ttn: A domain-shift aware batch normalization in test-time adaptation. *arXiv preprint arXiv:2302.05155*, 2023.

[27] Kuang Liu. `pytorch-cifar`: 95.47% on cifar10 with pytorch. https://github.com/kuangliu/pytorch-cifar, 2018. Accessed: 2025-08-28.

[28] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *2015 IEEE International Conference on Computer Vision (ICCV)*, page 3730–3738. IEEE, December 2015. doi: 10.1109/iccv.2015.425. URL http://dx.doi.org/10.1109/ICCV.2015.425.

[29] Aaron Lou, Isay Katsman, Qingxuan Jiang, Serge Belongie, Ser-Nam Lim, and Christopher De Sa. Differentiating through the fréchet mean. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[30] Stanislav Minsker. Geometric median and robust estimation in banach spaces. *Bernoulli*, 21(4), November 2015. doi: 10.3150/14-bej645. URL https://doi.org/10.3150/14-bej645.

[31] Zachary Nado, Shreyas Padhy, D Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*, 2020.

[32] Hongseok Namkoong and John C Duchi. Stochastic gradient methods for distributionally robust optimization with f-divergences. *Advances in neural information processing systems*, 29, 2016.

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[34] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in neural information processing systems*, 29, 2016.

[35] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. URL https://arxiv.org/abs/1801.04381.

[36] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL

https://proceedings.neurips.cc/paper_files/paper/2018/file/905056c1ac1dad141560467e0a99e1cf-Paper.pdf.

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. URL https://arxiv.org/abs/1409.1556.

[38] Cecilia Summers and Michael J Dinneen. Four things everyone should know to improve batch normalization. *arXiv preprint arXiv:1906.03548*, 2019.

[39] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/tan19a.html.

[40] Jeremy Watt, Reza Borhani, and Aggelos K. Katsaggelos. *Machine Learning Refined: Foundations, Algorithms, and Applications*. Cambridge University Press, 2 edition, 2020. doi: 10.1017/9781108690935.

[41] Shuang Wu, Guoqi Li, Lei Deng, Liu Liu, Dong Wu, Yuan Xie, and Luping Shi. $l1$-norm batch normalization for efficient training of deep neural networks. *IEEE transactions on neural networks and learning systems*, 30(7):2043–2051, 2018.

[42] Yuxin Wu and Kaiming He. Group normalization, 2018.

[43] Hongwei Yong, Jianqiang Huang, Xiansheng Hua, and Lei Zhang. Gradient-centralization: A new optimization technique for deep neural networks. 2020.

[44] Xiaoyong Yuan, Zheng Feng, Matthew Norton, and Xiaolin Li. Generalized batch normalization: Towards accelerating deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1682–1689, 2019.

[45] Runtian Zhai, Chen Dan, Zico Kolter, and Pradeep Ravikumar. Doro: Distributional and outlier robust optimization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12345–12355. PMLR, 18–24 Jul 2021. URL http://proceedings.mlr.press/v139/zhai21a.html.

[46] Banghua Zhu, Jiantao Jiao, and Jacob Steinhardt. Generalized resilience and robust statistics. *The Annals of Statistics*, 50(4), August 2022. ISSN 0090-5364. doi: 10.1214/22-aos2186. URL http://dx.doi.org/10.1214/22-AOS2186.

## Appendix A. Proof of Theorem 3.1

We prove that the second derivative of some generalized 1-dimensional geometric median

$$g(y) = \sum_{i=1}^{m} |x_i - y|^{\phi} \tag{6}$$

will always be convex for $\phi \geq 1$ by showing that the second derivative of $g(y)$ with respect to $y$ is strictly non-negative. The first derivative of $g(y)$ is

$$\frac{\partial g}{\partial y} = \sum_{i=1}^{m} \phi |x_i - y|^{\phi - 1} \cdot \frac{x_i - y}{|x_i - y|}, \tag{7}$$

$$\tag{8}$$

however,

$$\frac{x_i - y}{|x_i - y|} = \frac{|x_i - y|}{x_i - y} \tag{9}$$

because $(x_i - y)^2 = |x_i - y|^2$. This gives us

$$\frac{\partial g}{\partial y} = \sum_{i=1}^{m} \phi |x_i - y|^{\phi - 1} \cdot \frac{|x_i - y|}{x_i - y} \tag{10}$$

$$= \sum_{i=1}^{m} \phi \frac{|x_i - y|^{\phi}}{x_i - y}. \tag{11}$$

The second derivative of $g(y)$ w.r.t. $y$ is

$$\frac{\partial^2 g}{\partial y^2} = \frac{\partial g}{\partial y} \left( \sum_{i=1}^{m} \phi \frac{|x_i - y|^{\phi}}{x_i - y} \right) \tag{12}$$

$$= \sum_{i=1}^{m} \phi \left( \frac{\phi |x_i - y|^{\phi}}{(x_i - y)^2} - \frac{|x_i - y|^{\phi}}{(x_i - y)^2} \right) \tag{13}$$

$$= \sum_{i=1}^{m} \frac{(\phi^2 - \phi)|x_i - y|^{\phi}}{(x_i - y)^2} \tag{14}$$

which is strictly positive when $\phi^2 - \phi$ is positive, which occurs when $\phi \geq 1$. Note that while $\phi < 0$ looks convex, the landscape of the 1-dimensional geometric median is not well-defined at points where $y = x_i$ when $\phi < 0$.

## Appendix B. Newton's Method

In this section, we discuss Newton's method, a local optimization scheme based on the second order Taylor series approximation that tends to perform well when minimizing convex functions of a moderate number of inputs [23, 40]. By incorporating both the gradient and Hessian, it often converges faster than first-order methods when the second derivative is available and inexpensive to compute [23, 40]. In AugNorm, the univariate case allows the second derivative to be computed

and inverted efficiently, motivating its use over alternatives such as gradient descent, tabu search, or simulated annealing.

At iteration $k$, Newton's update is given by the stationary point of the quadratic approximation around $\mathbf{y}^{k-1}$:

$$\mathbf{y}^k = \mathbf{y}^{k-1} - \left(\nabla^2 g(\mathbf{y}^{k-1})\right)^{-1} \nabla g(\mathbf{y}^{k-1}). \tag{15}$$

For the univariate case, this simplifies to the familiar update rule:

$$y^k = y^{k-1} - \frac{F_Y(y^{k-1})}{F_{YY}(y^{k-1})}. \tag{16}$$

In the AugNorm setting, each iteration takes the form:

$$y^k = y^{k-1} - \frac{\sum_{i=1}^m \phi \frac{|x_i - y^{k-1}|^\phi}{x_i - y^{k-1}}}{\sum_{i=1}^m \frac{(\phi^2 - \phi)|x_i - y^{k-1}|^\phi}{(x_i - y^{k-1})^2}}. \tag{17}$$

## Appendix C. Theorem 4.2 and Proof

**Theorem 2** *When backpropagating through AugNorm on inputs $\mathbf{X} \in \mathbb{R}^{m \times d}$ ($m$ batched inputs with dimension(s) $d$) we get the following derivatives:*

$$\frac{\partial \ell}{\partial \hat{x}_{ij}} = \frac{\partial \ell}{\partial y_{ij}} \gamma_j \tag{18}$$

$$\frac{\partial \ell}{\partial \sigma_j^2} = \sum_{i=1}^m \frac{\partial \ell}{\partial \hat{x}_{ij}} \cdot -\frac{x_{ij} - \mu_j}{2(\sigma_j^2 + \epsilon)^{\frac{3}{2}}} \tag{19}$$

$$\frac{\partial \ell}{\partial s_{ij}} = \frac{\partial \ell}{\partial \sigma_j^2} \frac{2}{m}(x_{ij} - \mu_j) + \frac{\partial \ell}{\partial \hat{x}_{ij}} \frac{1}{\sqrt{\sigma_j^2 + \epsilon}} \tag{20}$$

$$\frac{\partial \ell}{\partial \mu_j} = -\sum_{i=1}^m \frac{\partial \ell}{\partial s_{ij}} \tag{21}$$

$$\frac{\partial \ell}{\partial x_{ij}} = \frac{\partial \ell}{\partial s_{ij}} + \frac{\partial \ell}{\partial \mu_j} \cdot \frac{-|x_{ij} - \mu_j|^{\phi - 2}}{\sum_{i=1}^m |x_{ij} - \mu_j|^{\phi - 2}} \tag{22}$$

$$\frac{\partial \ell}{\partial \gamma} = \sum_{i=1}^m \frac{\partial \ell}{\partial y_{ij}} \hat{x}_{ij} \tag{23}$$

$$\frac{\partial \ell}{\partial \beta_j} = \sum_{i=1}^m \frac{\partial \ell}{\partial y_{ij}} \tag{24}$$

**Proof** We use $M_\beta$ when referring to the minima along the generalized geometric median function and $y$ as the output to the normalization scheme (similar to the original batch normalization paper where $y = \gamma x + \beta$) [20].

Starting with the easiest derivations,

$$\frac{\partial \ell}{\partial \hat{x}_i} = \frac{\partial \ell}{\partial y_i}\frac{\partial y_i}{\partial \hat{x}_i} \tag{25}$$

$$= \frac{\partial \ell}{\partial y_i} \cdot \gamma \tag{26}$$

The derivatives of loss with respect to affine parameters do not change:

$$\frac{\partial \ell}{\partial \beta} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i}\frac{\partial y_i}{\partial \beta} \tag{27}$$

$$= \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i} \tag{28}$$

and

$$\frac{\partial \ell}{\partial \gamma} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i}\frac{\partial y_i}{\partial \gamma} \tag{29}$$

$$= \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i}\hat{x}_i \tag{30}$$

Next, we compute $\frac{\partial \ell}{\partial \sigma_\beta^2}$ and $\frac{\partial \ell}{\partial M_\beta}$ as follows.

$$\frac{\partial \ell}{\partial \sigma_\beta^2} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i}\frac{\partial \hat{x}_i}{\partial \sigma_\beta^2} \tag{31}$$

$$= \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i} \cdot (x_i - M_\beta) \cdot \frac{-1}{2}(\sigma_\beta^2 + \epsilon)^{-3/2} \tag{32}$$

$$\frac{\partial \ell}{\partial M_\beta} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i}\frac{\partial \hat{x}_i}{\partial M_\beta} + \frac{\partial \ell}{\partial \sigma_\beta^2}\frac{\partial \sigma_\beta^2}{\partial M_\beta} \tag{33}$$

$$= \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i}\frac{-1}{\sqrt{\sigma_\beta^2 + \epsilon}} + \frac{\partial \ell}{\partial \sigma_\beta^2}\frac{2(x_i - M_\beta)}{m} \tag{34}$$

These equations remain mostly the same for both classical batch normalization and augmented batch normalization, with the small caveat that the mean $\mu_\beta$ is replaced with the generalized geometric median minima $M_\beta$. These prior computations also allow us to calculate $\frac{\partial \ell}{\partial x_i}$.

Again from [20], we know that

$$\frac{\partial \ell}{\partial x_i} = \frac{\partial \ell}{\partial \hat{x}_i}\frac{\partial \hat{x}_i}{\partial x_i} + \frac{\partial \ell}{\partial \sigma_\beta^2}\frac{\partial \sigma_\beta^2}{\partial x_i} + \frac{\partial \ell}{\partial M_\beta}\frac{\partial M_\beta}{\partial x_i} \tag{35}$$

$$= \frac{\partial \ell}{\partial \hat{x}_i}\frac{1}{\sqrt{\sigma_\beta^2 + \epsilon}} + \frac{\partial \ell}{\partial \sigma_\beta^2}\frac{2(x_i - M_\beta)}{m} + \frac{\partial \ell}{\partial M_\beta}\frac{\partial M_\beta}{\partial x_i}, \tag{36}$$

concluding all core derivatives for AugNorm outside of the gradients for the generalized geometric median. The gradient $\frac{\partial M_\beta}{\partial x_i}$ is computed in Appendix C.1. ■

### C.1. Differentiating through the Generalized Geometric Median

**Lemma 3** *Let $f : \mathbb{R}^m \to \mathbb{R}$ be a continuous function with first and second derivatives. Let $g(\mathbf{x}) = \arg\min_y f(\mathbf{x}, y)$, $\mathbf{x} = (x_1, \ldots, x_m)$, with $x_1, \ldots, x_m$ all independent, and $y \in \mathbb{R}$. Then:*

$$\frac{\partial g(\mathbf{x})}{\partial x_i} = -\frac{f_{X_i Y}(\mathbf{x}, g(\mathbf{x}))}{f_{YY}(\mathbf{x}, g(\mathbf{x}))} \tag{37}$$

*where $f_{X_i Y} = \frac{\partial^2 f}{\partial x_i \partial y}$ and $f_{YY} = \frac{\partial^2 f}{\partial y^2}$*

**Proof** We follow a similar proof structure as [11] to derive $\frac{\partial g(\mathbf{x})}{\partial x_i}$ To begin, we have

$$\frac{\partial f(\mathbf{x}, y)}{\partial y}\Big|_{y=g(\mathbf{x})} = 0 \tag{38}$$

so,

$$\frac{d}{dx_i}\frac{\partial f(\mathbf{x}, g(\mathbf{x}))}{\partial y} = 0. \tag{39}$$

Via chain rule, we also have:

$$\frac{d}{dx_i}\frac{\partial f(\mathbf{x}, g(\mathbf{x}))}{\partial y} = \frac{\partial^2 f(\mathbf{x}, g(\mathbf{x}))}{\partial x_i \partial y} \cdot \frac{dx_i}{dx_i} + \sum_{j \neq i, 1 \leq j \leq m} \left(\frac{\partial^2 f(\mathbf{x}, g(\mathbf{x}))}{\partial x_j \partial y} \cdot \frac{dx_j}{dx_i}\right) + \frac{\partial^2 f(\mathbf{x}, g(\mathbf{x}))}{\partial^2 y} \cdot \frac{\partial g(\mathbf{x})}{\partial x_i} \tag{40}$$

$$= \frac{\partial^2 f(\mathbf{x}, g(\mathbf{x}))}{\partial x_i \partial y} + \frac{\partial^2 f(\mathbf{x}, g(\mathbf{x}))}{\partial^2 y} \cdot \frac{\partial g(\mathbf{x})}{\partial x_i} \tag{41}$$

Equating this expression to 0 and rewriting in terms of $\frac{\partial g(\mathbf{x})}{\partial x_i}$ yields:

$$\frac{\partial g(\mathbf{x})}{\partial x_i} = -\left(\frac{\partial^2 f(\mathbf{x}, g(\mathbf{x}))}{\partial^2 y}\right)^{-1} \frac{\partial^2 f(\mathbf{x}, g(\mathbf{x}))}{\partial x_i \partial y} \tag{42}$$

$$= -\frac{f_{X_i Y}(\mathbf{x}, g(\mathbf{x}))}{f_{YY}(\mathbf{x}, g(\mathbf{x}))} \tag{43}$$

■

To compute $\frac{\partial \mu_j}{\partial x_{ij}}$ and $\frac{\partial \mu_j}{\partial \phi_j}$, we use Lemma A.1. We define a function $g(\mathbf{x}, \phi)$ for $\mathbf{x} \in \mathbb{R}^m$, $\phi \in \mathbb{R}$ such that

$$g(\mathbf{x}, \phi) = \arg\min_{\mu} f(\mathbf{x}, \phi, \mu) \tag{44}$$

where

$$f(\mathbf{x}, \phi, y) = \sum_{i=1}^{m} |x_i - y|^{\phi} \tag{45}$$

and $y \in \mathbb{R}$. Note that when $\phi = 1$ or $\phi = 2$, analytical solutions exist and they are represented by the median and mean, respectively.
First, we compute $f_{ZZ}(\mathbf{x}, \phi, g(\mathbf{x}, \phi)) \in \mathbb{R}$, which is known from Theorem 3.1:

$$\sum_{i=1}^{m} \frac{(\phi^2 - \phi)|x_i - g(\mathbf{x}, \phi)|^{\phi}}{(x_i - g(\mathbf{x}, \phi))^2} \tag{46}$$

Then we compute $f_{X_i Z}(\mathbf{x}, \phi, g(\mathbf{x}, \phi)) \in \mathbb{R}$: Starting with

$$f_Z = \sum_{i=1}^{m} \phi \frac{|x_i - g(\mathbf{x}, \phi)|^\phi}{x_i - g(\mathbf{x}, \phi)}, \tag{47}$$

we then obtain:

$$f_{X_i Z} = \phi \left( \frac{\phi |x_i - g(\mathbf{x}, \phi)|^\phi}{(x_i - g(\mathbf{x}, \phi))(g(\mathbf{x}, \phi) - x_i)} - \frac{|x_i - g(\mathbf{x}, \phi)|^\phi}{(x_i - g(\mathbf{x}, \phi))(g(\mathbf{x}, \phi) - x_i)} \right) \tag{48}$$

$$= \frac{(\phi - \phi^2)|x_i - g(\mathbf{x}, \phi)|^\phi}{(x_i - g(\mathbf{x}, \phi))^2}. \tag{49}$$

Finally we have $\frac{\partial g(\mathbf{x}, \phi)}{x_i}$:

$$\frac{\partial g(\mathbf{x}, \phi)}{\partial x_i} = \frac{\frac{(\phi - \phi^2)|x_i - g(\mathbf{x}, \phi)|^\phi}{(x_i - g(\mathbf{x}, \phi))^2}}{\sum_{i=1}^{m} \frac{(\phi^2 - \phi)|x_i - g(\mathbf{x}, \phi)|^\phi}{(x_i - g(\mathbf{x}, \phi))^2}} \tag{50}$$

$$= \frac{\frac{-|x_i - g(\mathbf{x}, \phi)|^\phi}{(x_i - g(\mathbf{x}, \phi))^2}}{\sum_{i=1}^{m} \frac{|x_i - g(\mathbf{x}, \phi)|^\phi}{(x_i - g(\mathbf{x}, \phi))^2}} \tag{51}$$

$$= \frac{-|x_i - g(\mathbf{x}, \phi)|^{\phi-2}}{\sum_{i=1}^{m} |x_i - g(\mathbf{x}, \phi)|^{\phi-2}} \tag{52}$$

$$\tag{53}$$

Note that at this point, it is sufficient to obtain the gradients if you utilize a fixed $\phi$ value (just let $g : \mathbb{R}^m \to \mathbb{R}$ instead of $g : \mathbb{R}^{m+1} \to \mathbb{R}$ by simply discounting the additional argument $\phi$). However, if you allow $\phi$ to be a model parameter (as is the case in the DiffPhi implementation of AugNorm), then you need to compute that additional gradient. For $\frac{\partial \mu_j}{\partial \phi_j}$, we first have

$$\frac{f_{YZ}}{f_{ZZ}} = \sum_{i=1}^{m} \frac{|x_i - g(\mathbf{x}, \phi)|^\phi (1 + \ln |x_i - g(\mathbf{x}, \phi)|)}{x_i - g(\mathbf{x}, \phi)} \tag{54}$$

so

$$\frac{\partial g(\mathbf{x}, \phi)}{\partial \phi} = \frac{\sum_{i=1}^{m} \frac{|x_i - g(\mathbf{x}, \phi)|^\phi (1 + \ln |x_i - g(\mathbf{x}, \phi)|)}{x_i - g(\mathbf{x}, \phi)}}{\sum_{i=1}^{m} \frac{(\phi^2 - \phi)|x_i - g(\mathbf{x}, \phi)|^\phi}{(x_i - g(\mathbf{x}, \phi))^2}} \tag{55}$$

$$alpa = \frac{\sum_{i=1}^{m} \frac{|x_i - g(\mathbf{x}, \phi)|^\phi (1 + \ln |x_i - g(\mathbf{x}, \phi)|)}{x_i - g(\mathbf{x}, \phi)}}{(\phi^2 - \phi) \sum_{i=1}^{m} |x_i - g(\mathbf{x}, \phi)|^{\phi-2}}. \tag{56}$$

## Appendix D. Dataset Summaries and Metrics

### D.1. Datasets

For CIFAR-10, we use the standard 80/20 train-test split, using fixed hyperparameters for each model taken from an online repository [27] and average over five random seeds. For subpopulation

Table 3: Number of training instances in each domain of CelebA.

| **CelebA** | Blond | Others |
|---|---|---|
| Male | 1387 | 66874 |
| Female | 22880 | 71629 |
| Young | 20230 | 106558 |
| Old | 4037 | 31945 |
| Attractive | 17008 | 66595 |
| Not-attractive | 7259 | 71908 |
| Straight-hair | 5178 | 28769 |
| Wavy-hair | 11342 | 40640 |
| Total | 162770 | |

shift we test on CelebA [28], a vision dataset with 162,770 training instances that is captured in the wild and labeled by humans to showcase real-world distribution shift. This dataset consists of human face recognition, where the target is whether the person has blond hair. We randomly sample 70% of the instances to be the training data and the rest is the validation/testing data (using official train-validation-test splits), averaging results over 10 fixed random seeds.

Additional specifics for the CelebA training data are included in Table 3.

### D.2. CIFAR10 Training Summary

For CIFAR10 image classification, five models were tested: ResNet-18 [14], MobiletNet-V2 [35], DenseNet121 [17], VGG19 [37], and EfficientNet-B0 [39]. These architectures were chosen because they represent a balance of CNNs that perform well on CIFAR10 while also being of high relevance.

For all models, we train for 200 epochs with momentum SGD using an initial learning rate of 0.1, momentum 0.9, weight decay of 0.0005, batch size of 128, and a Cosine annealing learning rate scheduler over all 200 epochs.

### D.3. CelebA Training Summary

As a backbone classification model ResNet18 was used. From [45], momentum SGD was used with learning rate 0.001, momentum 0.9, weight decay 0.001, and a batch size of 400. The hyperparameters used were $\alpha = 0.1$ for CVaR, $\alpha = 0.2, \epsilon = 0.005$ for CVaR-DORO, $\alpha = 0.25$ for $\chi^2$-DRO, $\alpha = 0.25, \epsilon = 0.01$ for $\chi^2$-DORO. For the 10 runs, seeds $\{0, 1, 2, \ldots, 9\}$ were used. It is worth noting that in some instances performance recorded from these seeds was worse than what was reported in [45], but we note that their code was pulled verbatim and the identical 10 seeds were used to record all results.

Additionally, for the robustness region bounded by $\rho$, we select $\rho = -\log(\alpha)$ when using CVaR or CVaR-DORO and $\rho = \frac{1}{2}(\frac{1}{\alpha} - 1)^2$ for $\chi^2$-DRO or $\chi^2$-DORO. For the divergence function $D$, we select $\sup \log \frac{dQ}{dP}$ for CVaR and CVaR-DORO, and $\int (dQ/dP - 1)^2 dP$ for $\chi^2$-DRO and $\chi^2$-DORO.

Table 4: Average test accuracies under different normalization methods on CelebA. Bold numbers indicate the best result per training procedure.

| **Method** | **BN** | **AN-1.5** | **DiffPhi** |
|---|---|---|---|
| ERM | $94.38 \pm 1.63$ | $95.01 \pm 0.37$ | $\mathbf{95.12 \pm 0.08}$ |
| ChiSq | $83.27 \pm 2.56$ | $83.40 \pm 2.74$ | $\mathbf{83.41 \pm 2.08}$ |
| ChiSq-DORO | $\mathbf{82.85 \pm 2.55}$ | $80.65 \pm 2.79$ | $81.52 \pm 3.34$ |
| CVaR | $83.70 \pm 1.56$ | $\mathbf{84.07 \pm 2.14}$ | $82.95 \pm 1.74$ |
| CVaR-DORO | $\mathbf{94.01 \pm 0.58}$ | $92.81 \pm 0.65$ | $93.73 \pm 1.00$ |

## Appendix E. CelebA Average Test Accuracy

The results in Table 4 report average test accuracies across subpopulations on CelebA. While AugNorm $\phi = 1.5$ and the differentiable $\phi$ variant (DiffPhi) achieve better performance on ERM and DRO-based frameworks, they tend to do worse in DORO-based training approaches (with DiffPhi doing markedly better under these conditions). We posit this is due to AugNorm biasing models towards worst-case scenarios, and thus while the worst-case accuracy tends to improve, average accuracy can fall off, especially if outlier-specific approaches (such as DORO) are already introduced to the training regime.

Combined with the worst-case accuracy upside reported in the main text, this indicates that AugNorm is still valuable in settings that experience subpopulation shift without sacrificing significant average accuracy.